AF INSTITUTE OF TECHNOLOGY

AN AIR BASE VULNERABILITY ASSESSMENT ANALYSIS TOOL FOR U.S. AIR FORCE WAR PLANNERS
VOLUME I: DEVELOPMENT AND USER'S MANUAL

THESIS

Richard M. Cockley
Captain, USAF

AFIT/GLM/LSM/90S-12

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

2

AFIT/GLM/LSM/90S-12

AN AIR BASE VULNERABILITY ASSESSMENT ANALYSIS TOOL FOR U.S. AIR FORCE WAR PLANNERS VOLUME I: DEVELOPMENT AND USER'S MANUAL

THESIS

Richard M. Cockley
Captain, USAF

AFIT/GLM/LSM/90S-12

Approved for public release; distribution unlimited

The opinions and conclusions in this paper are those of the author and are not intended to represent the official position of the DOD, USAF, or any other government agency.

AFIT/GLM/LSM/90S-12

AN AIR BASE VULNERABILITY ASSESSMENT ANALYSIS TOOL FOR U.S. AIR FORCE WAR PLANNERS

VOLUME I: DEVELOPMENT AND USER'S MANUAL

THESIS

Presented to the Faculty of the School of Systems and

and Logistics of the Air Force Institute of Technology

Air University

In Partial Fulfillment of the

Requirements for the Degree of

Master of Science in Logistics Management

Richard M. Cockley

Captain, USAF

September 1990

Approved for public release; distribution unlimited

Preface

The purpose of this study was to improve the pre- and post-processor BasePlot before distributing it to experienced Air Base Operability (ABO) analysts and planners. BasePlot was capable of pre- and post-processing TSARINA's (Theater Simulation of Air Base Resources INputs using AIDA) inputs and outputs but it needed to be modified, enhanced, documented and assessed before it could be distributed. The accomplishment of this goal, allows ABO analysts experienced in the use of TSARINA and a knowledge of ABO planning to observe on screen the results of an attack scenario run in TSARINA.

Credit for any success in achieving this goal must, however, be shared with the many fine people who helped me throughout this effort.

I am indebted to my thesis advisor Major Diener. He provided invaluable guidance, inspiration, and continual support. I am grateful to Lt Col Halliday for suggesting this project and for providing technical guidance and assistance. Finally, I owe the greatest debt to my wife Dennette, for her deep love, support, and understanding while I was devoted to this study.

Richard M. Cockley

Table of Contents

Page

List of Figures

Abstract

The purpose of this study was to improve the pre- and post-processor BasePlot before distributing it to experienced Air Base Operability (ABO) analysts and planners.

The Author was able to enhance, modify, and document BasePlot to meet the needs of ABO analysts and planners using QuickBASIC 4.5. The application will be implemented upon distribution.

Volume I contains four chapters and an appendix. Chapter I is the introduction containing the general issue, a statement of the specific problem, investigative questions, background of current issues on ABO, history and capability of TSAR and TSARINA, comparison to comparable software packages, BasePlot's capabilities, and a discussion of the current philosophies of documentation development. Chapter II details the thesis methodology used to solve the problem. This chapter describes software modification and testing, documentation, user testing, validation, and implementation. Chapter III describes the design, development, and outcome of the modified and enhanced BasePlot. Chapter III also provides an analysis of the pre- and post-processor BasePlot with respect to the goal of creating an efficient software package. Chapter IV describes the results of BasePlot's application and its

strengths and weaknesses. This chapter also provides recommendations for follow-on research. The Appendix is Baseplot's User's Manual which documents BasePlot's operations and provides examples of how to use the software package.

AN AIR BASE VULNERABILITY ASSESSMENT ANALYSIS

TOOL FOR U.S. AIR FORCE WAR PLANNERS

VOLUME 1: DEVELOPMENT AND USER'S MANUAL

I. Introduction

General Issue

The improvements made by our adversaries in their ability to deliver conventional and chemical weapons forces air base operability (ABO) analysts to take a critical look at the ability of our air bases to survive and recover from an attack. Former Secretary of Defense Caspar Weinberger emphasized the importance of ABO in a report to Congress, in which he stated, "We must be prepared for increased exposure of more NATO air bases to more enemy aircraft and missiles, with less time to prepare against attack" (Ray, 1985:34). This increased threat highlights the importance of ABO and with upcoming budget reductions these issues will be brought to the forefront of the budget debates on Capitol Hill.

These budget debates will affect the ability of our air bases to survive and recover from an attack. The importance ABO plays in the future depends on the ability of the ABO program to meet its objectives with reduced budgets. These objectives have been referred to as the pillars of ABO (refer to Figure 1 on the next page). The four pillars of ABO are defense, survival, recovery, and generation.

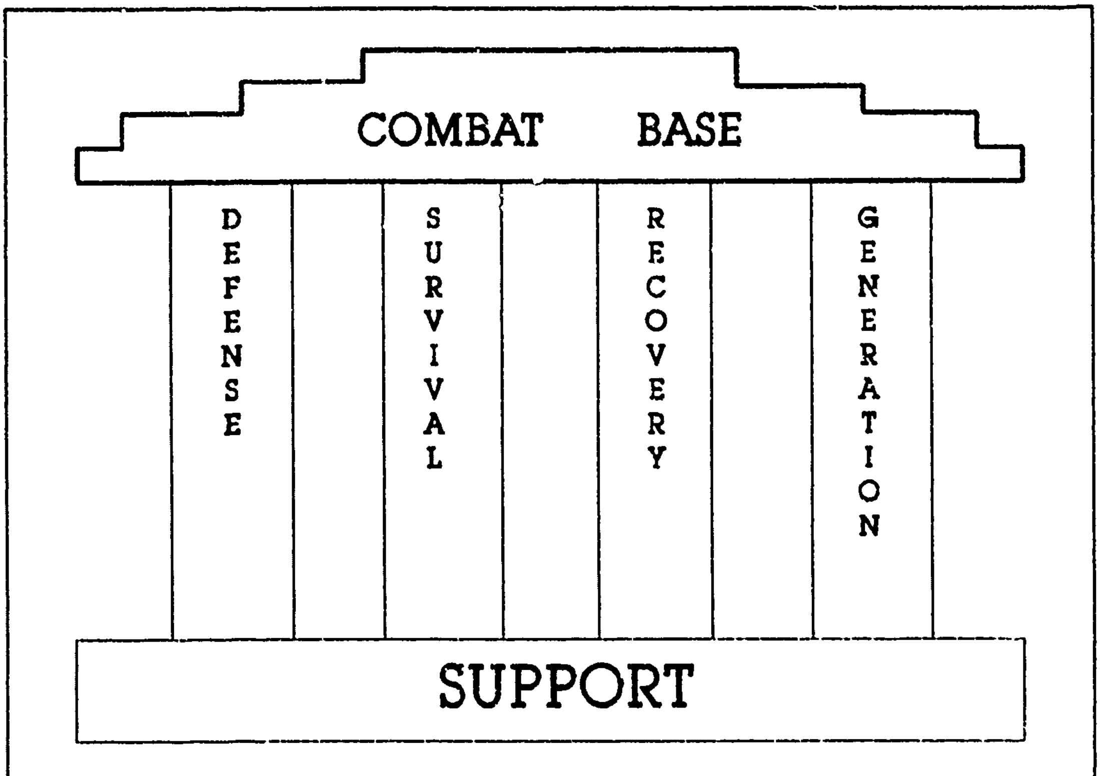

Figure 1. Pillars of ABO (Huelskamp, 1989)

Defense includes area air defense, point air defense, and air base ground defense. Survival incorporates such things as hardening shelters, dispersing shelters, camouflage, concealment, and deception. The recovery pillar includes making essential repairs to the base facilities that affect the launching and recovery of aircraft. The final pillar of ABO, generation, ensures the base is able to continue to generate sorties and includes systems for command, control, communications, and computers (SC4) which ensures that command and control can be maintained during an attack and recovery from an attack (Ray, 1985:36).

The increased range and sophistication of our adversaries has increased the number of bases and attack scenarios the ABO analyst must consider. ABO analysts use a Monte Carlo computer simulation model (Emerson, 1985a:1) to assess an air base's vulnerability to an enemy's conventional or chemical attack. The Theater Simulation of Air base Resources INputs using AIDA (TSARINA) model is run on a main-frame or micro-computer and requires the user to have an extensive working knowledge of ABO.

TSARINA uses input target cards (which place shelters and other facilities on the base using X-Y coordinates) and input attack cards (which provide a simulated attack using 10 different weapon types) to simulate the damage that an attack would cause to base shelters, facilities and resources. The damage is based on the impact points of the weapons. These impact points are represented using an X-Y coordinate and a radius of effect (the effect radius is represented by a circle within which damage would occur to resources).

With these cards, TSARINA provides the ABO analyst with the ability to model a large spectrum of wartime environments but it was not designed to do any graphic pre- or post-processing. Graphics pre- and post-processors enable computer data to be viewed on a computer screen. Pre- and post-processing TSARINA's inputs and outputs, allows the ABO analysts to view their simulated attacks on the computer screen. This allows the ABO analyst to observe on the screen the effect the attack has on the base structures and resources (O'Neill, 1989).

There is a program capable of pre- and post-processing TSARINA's inputs and outputs but it needs to be modified, refined, and documented. The program, BasePlot, needs to be refined and modified because it was designed to be used by the programmer and not by other users. This caused BasePlot to be very unforgiving of incorrect user inputs. BasePlot includes nine different options for assessing simulated air base attack scenarios and thus determining the vulnerability of that air base.

BasePlot was developed by Capt Bob O'Neill, AFCSA/SAGO, using Quickbasic 4.5 and it is run on a microcomputer. BasePlot allows analysts experienced in the use of TSARINA and a knowledge of ABO planning to observe on screen the results of an attack scenario run in TSARINA.

Specific Problem

The objective of this thesis is to modify, enhance, document, and assess BasePlot before distributing it to experienced ABO analysts and planners.

Investigating Questions

In order to meet the objective of modifying, enhancing, documenting and assessing BasePlot, a graphics pre- and post-processor, the following investigative questions need to be answered:

1. What enhancements/modifications need to be made to BasePlot before it can be distributed to ABO analysts?

a. What improvements or added capabilities need to be added?

b. What level of user-friendliness is needed?

2. What type of assessment needs to be performed on this program?

a. Does the existing program need to be modified?

b. Is the current computer program valid and who should test its validity?

c. Does the program follow conventional structured programming philosophies?

3. What documentation is required before the software can be distributed to ABO analysts?

a. What type of documentation is required?

b. Will the user require a computer-aided tutor for the program designed to be run on the personal computer to enhance any hard copy documentation?

Background

Introduction. The background begins by discussing the different pillars of ABO and why ABO is important. Covered next is a brief history of TSAR and TSARINA and their capabilities, followed by a review of research studies that address the validity of TSAR and TSARINA. Next, there is a discussion of graphics pre- and post-processors similar to BasePlot that are available and why they are not being used by Air Force analysts. A specific description of BasePlot and its capabilities, is covered next. Finally, the background concludes with a look at the current documentation philosophies.

Air Base Operability. The ability to achieve and maintain air superiority is the main goal of the air forces in NATO (Ray, 1985:34). The easiest way to achieve air superiority is to eliminate the enemies' ability to launch and recover aircraft. This strategy is addressed by Colonel Darrell G. Bittle, in "Air Base Survivability," which cites Italian Air Marshall Guilio Douhet as saying, in 1921, "it is more effective to destroy the enemy's aerial power by destroying his nest and eggs on the ground than to hunt his flying birds in the air (Bittle, 1981:7)." Considering this philosophy, it is easy to understand why Soviet doctrine emphasizes destroying the air fields of Western Europe (Dilullo, 1989:112). Our current war plans will not be effective if we cannot launch and recover our aircraft. To achieve this launch and recovery capability we need to improve our air base defense (Ray, 1985:34). In addition to the problem of launch and recovery is the demand our sophisticated aircraft place on ABO. These aircraft need long, smooth, and high-load-bearing runways due to their great weight and takeoff speeds (Nordeen, 1987:31). These and other problems compound the ability of ABO analysts to simulate our air base's vulnerability. Before continuing with the problems of ABO, a common understanding of its definition is required.

ABO was known as air base survivability (ABS) until 1983 when it was changed to include the ability to recover from attack as quickly as possible. Until this time the ABS program contained only those elements pertaining to surviving a hostile attack. This change increased the number of components in the definition of ABO. ABO is now defined as:

> an advocacy program aimed at improving our ability to survive and recover in a degraded environment and it includes those measures of active defense; base recovery; command, communications and control; and aircraft mods or enhancements necessary to defend against, mitigate the effects of, and recover from hostile enemy actions to achieve a sustained launch and recovery capability for effective combat operations. (Speigel, 1983:19)

The individual components (or pillars) of ABO include defense, survival, recovery, and generation. These pillars were previously presented but they need to be explained further at this point (refer to Figure 1). The defense is comprised of security radars, air interdiction aircraft, SAMs, point air defense (PAD), and air base ground defense (ABGD) (Ray, 1986:36). The survival pertains to those actions taken by the base to deter the ability of the attacking force from destroying fixed facilities and resources. The survival pillar includes "hardening of shelters, protective construction, nuclear, biological, and chemical (NBC) defense capabilities, camouflage, concealment, and deception" (Nordeen, 1987:32-33). Recovery, the next pillar of ABO, includes assessment, rapid runway repair (RRR), disposal of unexploded ordinance (UXO), decontamination, and repair of critical facilities (Nordeen, 1987:34). The final pillar of ABO, generation, ensures the base is able to continue to generate sorties and includes SC4 which ensures that command and control can be maintained during an attack and recovery from an attack (Ray, 1985:36).

The pillar that applies the most to TSARINA and BasePlot is survival. TSARINA provides the ability to perform simulated attacks on a base and review the effect of the attack on the facilities and resources. BasePlot allows the ABO analyst to observe on the screen the results of the attack. Together the two allow ABO analysts to make recommendations on the expected damage to a particular base, based on a particular attack scenario.

The ABO components can best be described as concentric circles of defense spread out from the base (Ray, 1985:36). The outer layer of circles consists of security radar systems which provide our first indication of an eminent attack. The next circle of defense is a combination of air interdiction aircraft and SAMs which provide the bulk of the air base's protection. Unfortunately these two systems will not be able to destroy all incoming aircraft, leaving point air defense as the final means of active defense to protect our air bases. Point air defense includes short range SAM systems called Rapier and Roland. The final concentric circle includes ABGD, an active defense measure, and the survival measures the base took prior to the attack.

Each of these components alone does not provide adequate protection, but together they achieve a greater synergistic survival rate. Although there are some critical problems associated with the ABO program, e.g., funding and policy issues (Besch, 1983:77-81), these components working together form a basis for providing adequate protection for air bases in the European theater.

TSAR & TSARINA. TSAR is a stochastic discrete event model (Emerson, 1985a:1; Law, 1982:4) that simulates the interactions among resources required by the activities necessary to generate aircraft sorties (Emerson, 1985a:1). It was developed because the existing simulation models were limited in their ability to simulate war time scenarios.

TSARINA is also a stochastic discrete event model that simulates conventional and chemical air base attacks (Emerson, 1985b:1). TSARINA can be used as a general-purpose damage assessment model or the outputs can be used in TSAR to check the impact of air base attacks on sortie generation (Emerson, 1985b:3).

Specifically, TSAR simulates the interaction of 11 different classes of resources: (1) aircraft, (2) aircrews, (3) ground personnel, (4) support equipment, (5) aircraft parts, (6) aircraft shelters, (7) munitions, (8) TRAP (Tanks, Racks, Adapters, and Pylons), (9) POL (Petroleum, Oil, and Lubricants), (10) building materials, and (11) air base facilities (Emerson, 1985a:1). TSAR, along with its companion model TSARINA, can simulate the impact of an air base attack and the recovery from such an attack (Emerson, 1985a:92). During these attacks, facilities, as well as other resources can be damaged and destroyed. Activities that require resources that are damaged or destroyed are either delayed, cancelled, or processed by an alternate set of resources (Emerson, 1985a:3). Post attack recovery includes the simulation of explosive ordnance disposal personnel clearing unexploded munitions from taxiways and runways as well as civil engineers performing emergency repairs to essential taxiways, runways, shelters and repair shops (Emerson, 1985a:117-123). TSAR/TSARINA also have the capability to model chemical warfare (CW) conditions (Emerson, 1985a:124). This includes the use of CW ensembles and their constraints (mobility, visibility, communications) on personnel while performing maintenance tasks (Emerson, 1985a:124-126).

TSARINA has been validated, but Emerson states that its validation is limited (Emerson, 1985b:6). This validation consisted of comparing TSARINA results with those of the accepted and validated Logistics Composite Model (LCOM) (Noble, 1986:1). Although no specific documentation of what comparisons were made is available, Emerson states that the results were "quite similar" (Emerson, 1985b:6). TSARINA has been compared to LCOM numerous other times and TSAR sortie generation simulations have, over time, become the "de facto AF standard ABO model" (Gornto, 1989:2). TSARINA was also compared to actual exercises (SALTY DEMO) which tested an air base's vulnerability. The results of the exercise attack damage was compared to TSARINA's simulated damage.

Comparison of Software. An extensive search of the literature revealed only two documented software packages available that are similar to BasePlot. One is the Combat Base Assessment Model (CBAM) and the second is a training tool used at the U.S. Air Force Academy called CRISIS.

CBAM is a deterministic, repeatable computational model (Huelskamp, 1988:2-35) that was designed to analyze the current major ABO concerns of the Air Force from a survivability perspective. It works in a microcomputer environment but it has very demanding hardware requirements. It can be used to rapidly evaluate alternative deployments, defensive tactics, and allocations of resources against a variety of likely conventional, unconventional, or chemical threats (Huelskamp, 1988:1-2).

CBAM is operational and has been shown to simulate attacks as effectively as TSARINA (Gornto, 1989:3). Although shown to work effectively, CBAM is designed as a stand alone simulation model and can not fairly be compared to Baseplot. However, the excessive restrictions CBAM places on the microcomputer make it incompatible with the majority of the microcomputers used by analysts (O'Neill, 1989). The final drawback, and the most important, is that CBAM is a very expensive software package and the AFCSA/SAGO office, who mostly do ABO analyses, are not interested in purchasing it (O'Neill, 1989).

CRISIS is used primarily by the civil engineers at the Air Force Academy as a training tool; it was not designed to be used by analysts in the field. CRISIS requires Computer-Aided Design and Drafting (CADD) to be run at the same time and it is very limited in its ability to perform ABO assessments in its current form. CRISIS performs three functions. It does base comprehensive planning, ABO combat support planning, and graphics programming showing critical wartime assets (Formwalt, 1989:ii). In order to perform these functions CRISIS includes the following five modules: 1) intro to CRISIS and CADD; 2) base comprehensive planning number one; 3) base comprehensive planning number two; 4) ABO pre-attack planning; and 5) ABO attack using Base Recovery After Attack (BRAAT).

The first module provides the user with an introduction to CRISIS and CADD. The second and third modules initiate a small scale design of a base. The ABO pre-attack includes preparing the base for an attack. The actions taken to prepare the base for an attack are the same as those mentioned in the survival pillar. They include camouflage, concealment, deception, hardening key facilities, evaluating assets, and dispersal. The final module uses a separate model called BRAAT to perform a simulated attack. BRAAT evaluates the damage, minimum operating strip (MOS), rapid runway repair (RRR), and sortie generation (Formwalt, 1989:5-1).

The two software packages differ from BasePlot in scope and capabilities but more importantly, neither program is intended to be used with TSARINA. Since TSARINA is the computer simulation model used extensively by ABO analysts these programs cannot be compared to BasePlot.

BasePlot. BasePlot was designed to allow analysts experienced in the use of TSARINA (Theater Simulation of Air Base Resources INputs using AIDA) and a knowledge of ABO planning to observe on screen the results of an attack scenario run in TSARINA. BasePlot was written by Captain Bob O'Neill, AFCSA/SAGO. It was edited and documented by Captain Richard Cockley, AFIT/LSG.

BasePlot is an easy to use graphic pre- and post-processor. It includes nine different options for assessing the simulated air base attack scenarios and thus determining the vulnerability of that air base. It also provides the ability to dump the screen contents to the a DataProducts color printer in order to get a hard copy of the simulated attack.

BasePlot uses primarily three ASCII text files in the TSARINA data format: the target file, the attack file, and the hit file. These text files can be existing TSARINA files or built from scratch by an ABO analyst. The user has the capability of adjusting the data within these text files as long as they understand TSARINA's data requirements.

BasePlot allows the ABO analyst to assess air base vulnerability with these current text files. After reviewing the results of the simulated attack the ABO analyst can make changes to the TSARINA attack file, make another run using TSARINA, and then view the resulting hit file to get another assessment of the air base's vulnerability.

BasePlot was originally developed by Capt Bob O'Neill to help a new ABO analyst understand the complexities of the data entered into and received from TSARINA. TSARINA simulates planned attacks (attack cards) on air base shelters, facilities, surfaces, and resources (target cards) to determine actual weapon impact points (hit cards) and the associated damage caused by these weapons.

The target cards contain the data needed to place rectangular representations of shelters, surfaces, and other facilities (subsequently these will be referred to as just facilities) on the base in an X-Y coordinate system. Each individual target card has an X-Y coordinate representing the lower left hand corner of the facility. The target card also includes the length and width in feet. The final piece of data TSARINA needs in order to complete the facility is the orientation (heading in degrees). Using this information, TSARINA computes the remaining three X-Y coordinates to complete the rectangular representation of the facility. If for example the orientation is 0 degrees, TSARINA places the upper left corner directly north of the

X-Y coordinate. The distance north, is equal to the width provided in the target card. The target card also specifies the target type (one of 30 different vulnerability categories), facility number, and other target descriptors.

The attack cards simulate an attack or series of attacks using up to ten different weapon types delivered by aircraft or surface-to-surface missiles. Each individual attack card contains the intended number of weapon-delivery passes (WDPs) an aircraft or missile would make on a base facility. This is identified by an aim point on the X-Y coordinate system. The next piece of information TSARINA needs is the heading, in degrees, of the WDP. The attack card continues with the number of bombs which will be dropped on that WDP and the length of the bomb stick associated with those bombs. The stick length varies depending on the aircraft delivery profile, the type of weapon, and the number of bombs in the WDP. For instance, a precision guided munition would have a stick length of zero while a drop of 10 general purpose bombs might have a stick length of 900 feet with 100 feet separation between each bomb. Finally, the attack card also specifies the type of weapon(s) in the WDP.

TSARINA normally produces statistical output showing the effect of the simulated attack on air base facilities. TSARINA can, in addition to statistical output, provide the ABO analyst with a hit file consisting of the X-Y coordinates of the individual bomb and unexploded ordinance

(UXO) impact points for each TSARINA trial. Using BasePlot, the ABO analyst can then view the hits by attack, by trial, or by some combination of both to determine which assets on the air base are at risk from this type of attack or see if a minimum operating strip (MOS) exists. Furthermore, the ABO analyst can overlay the impact points on top of the aim points (attack data) to see how randomness and uncertainty affect the outcome of an attack. The ABO analyst can also assess if TSARINA adequately models the accuracy and reliability of the weapons.

Documentation. Good software documentation is one key to a software's success. Documentation makes users satisfied, self-sufficient, and productive (Odescalchi, 1986:16). In order to make BasePlot a more widely used product it needs to be internally and externally (user's manual) documented.

There are many definitions of documentation but for purposes of clarification the following will suffice. "Documentation is organized, descriptive information concerning purposes, methods, logic, relationships, capabilities, and limitations of the software package" (Guillemette, 1990:69). One of the biggest shortfalls in software development in recent years has been in the development of the documentation. Documentation is mainly written by programmers as an afterthought and the documentation is often filled with jargon and technical information that most users do not understand. Users view the system from the outside and are concerned with performing the task at hand, not with how the computer program completes the task.

Internal documentation is primarily used by programmers. Therefore, the internal documentation needs to be technical and provide the programmer with the information they need to update or modify the software. This internal documentation allows for a historical track record, easier updates, and some user maintenance. The internal documentation should explain the various components of the software's code in enough detail so that updates can be made without having to spend an inordinate amount of time figuring out what the computer code is doing.

The primary focus of the documentation for BasePlot was geared toward developing a user's manual that could be used by an inexperienced ABO analyst. The user's manual is often the primary interface when difficulties occur in using software and is therefore a major part of the systems image (Guillemette, 1990:70). The user's manual should be readable, useable, and accessible. The readability of a user's manual is judged by its legibility, interest, and understanding (Guillemette 1987:38). A user's manual is judged to be useable if users can achieve the objectives they intended to do and the accessibility of a user's manual is based on the ability of users to find the information they are looking for (Guillemette, 1987:38).

One of the major problems with user's manuals is that they are based on faulty assumptions about their intended audience. Everyone recognizes that documentation is necessary, but few willingly take the responsibility of making it accurate, complete, organized, readable, and suitable for those using it (Fingerhut, 1986:19).

As an example, one faulty assumption made is that the reader only needs the information on how to enter the data. In reality, most users approach a new system with uncertainty and need reassurance that they are entering the proper data. This is especially true of the intended audience of BasePlot because they are inexperienced in using TSARINA and they need help getting BasePlot to work properly. Some of the other faulty assumptions include: 1) the audience already has key information; 2) documentation can overcome the system's design weaknesses; 3) steps need not be repeated once covered; 4) the reader understands technical jargon and acronyms; and 5) everything will go right (Fingerhut, 1986:19).

With these shortfalls in mind a user's manual should be developed using four major steps. These steps are defining the goal, identifying the audience, establishing a time line, and testing and reviewing the manual (Kraft, 1988:62-64). The goal should clearly indicate for whom the user's manual is being written and why. Identifying the audience includes examining their background and experience, computer literacy, use of the manual, and equipment configurations.

The time line is based on completing the manual in time to allow for testing and review. Testing is accomplished by giving a test group some sample questions which require using the user's manual and observing their reactions.

Another consideration that should be given toward development of a user's manual is the method in which it is oriented. There are two methods a manual can be oriented. The manual can be task-oriented (how to perform step-by-step) or product-oriented (describes how the system works) (Odescalchi, 1986:16). Because today's software packages are falling into the hands of inexperienced users, most manuals need to have the step-by-step orientation to allow the users to get the job done in the most productive and efficient manner. In a study performed by Stephen Partridge it was found that 79% of the participants preferred task-oriented software packages (Odescalchi, 1986:16).

Finally, user-friendliness needs to be considered by the developer of a software package when designing the user's manual. The ease with which a user can operate the software is important but so is the ease with which a user can use the manual. The following things need to be considered when making a user's manual user-friendly: 1) table of contents; 2) level of user knowledge; and 3) accurate and complete indexing (Martin, 1986:20).

The level of user knowledge pertains to writing the manual so that both inexperienced and experienced users can use the manual to perform the tasks they need without hindering the other. The manual should provide quick synopses of the functions without detail for the experienced user and in depth coverage of the same topic for the inexperienced user. This will allow the experienced user the ability to quickly find what they need without having to dig through the detailed descriptions.

Conclusion. The improvements made in the ABO program the last three years are significant. With daily improvements in SAM technology and the possibility of using versions of air-to-air missiles in a surface-to-air mission role (Correll, 1983:43), it appears that future ABO programs will provide better defense systems, thereby increasing U.S air base protection. These improvements will be required as our adversaries improve their capability to evade our current defenses.

This background defined the ABO program by examining the pillars of ABO and has shown why ABO is important. A brief history of TSAR and TSARINA was provided, followed by a review of the cases showing TSARINA's validity. Next, software similar to Baseplot was reviewed to show their capabilities and that neither one meets the requirements of the ABO analyst. This was followed by a discussion of BasePlot and its capabilities. Finally, the background took a brief look at the current philosophies of documentation and the approach that was taken in the development of the user's manual for BasePlot. The importance of ABO cannot be overlooked because the goal of obtaining air superiority will be severely jeopardized without the ability to launch and recover aircraft.

Scope and Limitations

This thesis results in a microcomputer based graphics pre- and post-processor with an accompanying user's manual and tutorial (demonstration). This program does not replace the simulation model TSARINA but enhances TSARINA by enabling an ABO analyst to view the inputs and outputs on the computer screen. This thesis does not deal with specific programing techniques or provide detailed background of the QuickBASIC programming language except where commands will be needed for future additions and maintenance.

Organization of the Thesis

This thesis contains four chapters and one appendix. Chapter I is the introduction containing the general issue, a statement of the specific problem, investigative questions, background of current issues on ABO, history and capability of TSAR and TSARINA, comparison of software packages, BasePlot's capabilities, and a discussion of the current philosophies of documentation development.

Chapter II details the thesis methodology used to solve the problem. This chapter describes software modification and testing, documentation, user testing, validation, and implementation.

Chapter III describes the design, development, and outcome of the modified and enhanced BasePlot. Chapter III also provides an analysis of the pre- and post-processor BasePlot with respect to the goal of creating an efficient software package.

Chapter IV describes the results of BasePlot's application and its strengths and weaknesses. This chapter also provides recommendations for follow-on research.

The Appendix is Baseplot's User's Manual which documents BasePlot's operations and provides examples of how to use the software package.

II. Methodology

Introduction

This chapter discusses the research methodology followed in developing the graphics pre- and post-processor called BasePlot into a software package capable of being distributed to ABO analysts. The process of modifying and testing BasePlot consisted of enhancing, testing, and assessing the current program by the developer, developing a user's manual, user testing the program and manual, program validation by an ABO faculty expert, and program implementation.

Program Enhancement

BasePlot was enhanced by adding some new capabilities and documenting the main program and all sub-programs. The original program was written in QuickBASIC 4.5 and to maintain consistency and software compatibility any new programming was done using QuickBASIC. The enhancement and testing of BasePlot was accomplished on a Zenith Z-248 microcomputer.

The first part of the enhancement included creating an initial menu system which brings the user from the MS-DOS operating system into BasePlot. This included allowing the user to enter the name of the air base they are simulating and asking for the target file for this air base. The target file contains the initial information BasePlot requires to display the air base on the computer screen. This new menu was also written using QuickBASIC. The new menu options were handled as separate program modules (sub-programs) to simplify the programming task. Rather than incorporate the new computer code necessary to process the user's inputs into the main program, the new options were written and debugged separately in a sub-program and which is called from the main program. The original program follows structured programming philosophies (this is discussed in the next section). The new options will maintain the structured programming procedures that where followed when the original program was written. There will be more discussion on structured programming techniques in Chapter III. As the modules were completed, the programmer tested them using actual data inputs. This allowed the programmer to make changes without affecting the main program.

The original program was designed to be used by the developer and as a result there was little consideration given to data inputs because the developer knew precisely what inputs were required. In its original form BasePlot is very unforgiving towards user's inputs. For example, if the user makes an incorrect data input (target filename entered incorrectly), the program indicates the type of error and terminates the program. This termination forces the user to start again from the DOS prompt.

Because of this, the next enhancement included the ability for BasePlot to notify the user of an incorrect entry made and allows them to make corrections to their entry before returning them to the previous menu option. This will enable users to feel more comfortable using the system knowing that they can make mistakes without interrupting the program.

The final step in enhancing the program included adding an introduction screen with a disclaimer statement. This screen shows the program title and program information. The introduction screen also provides the user with a point of contact and phone number to call if they run into any problems. The disclaimer removes the analysis responsibility from the programmer and places it on the user.

Program Testing and Assessment

The air base data files (target and attack) are used by BasePlot to provide a graphical picture of an air base on the computer screen. The target data file contains the TSARINA formatted target cards. The target cards describe shelters and other facilities on the base using X-Y coordinates. BasePlot takes the X-Y coordinate given and computes the remaining three X-Y coordinates to complete a rectangular representation of the shelter or facility.

The ABO analyst can then look at the facilities and verify that they were entered correctly on the target cards.

For instance, an ABO analyst intended to enter a runway to be 12,000 feet long and upon review of the targets on the computer screen it is found that it is only 1200 feet long. The analyst can now go back and change the target card that had the runway data on it. The target cards were compared to the computer screen to ensure that all facilities and paved surfaces were placed correctly. This tested BasePlot's plotting capabilities.

The attack cards simulate the attack using up to ten different weapon types. The individual attack card contains the intended weapon-delivery pass (WDP) an aircraft would make on a base resource. There is an X-Y coordinate which represents aim point of the WDP. The ABO analyst can look at their attack data cards (BasePlot can represent a maximum of 100 attacks) and visually check to see that the resources they intended to attack were in fact attacked. The information on the attack cards was tested in the same way as the target cards. Any discrepancies noticed can be fixed as well.

BasePlot was also assessed on its ability to follow the structured programming philosophies. The main goal of structured programming is to break a complicated programming task into smaller parts and thus make the overall programming task simpler. "It is much easier to solve a collection of smaller tasks than one large task" (Feldman, 1988:375). This idea is the main premise behind modular structured programming. The smaller tasks are called sub-programs or sub-functions and operate separately from the main program.

There are certain rules to follow when writing a structured program. The number of rules varies depending on the test being used but the overall philosophies are maintained. Different software programs have different programming capabilities and therefore since BasePlot was written in QuickBASIC, its programming structure was assessed using the following rules from a QuickBASIC programming text: 1) perform detailed tasks in the sub-programs or sub-functions; 2) the sub-programs or sub-programs should be easy to understand and maintain; 3) use top-down flow control; 4) constrain the programming procedures to the three basic structured constructs (assignment and computation, IF ... THEN or SELECT CASE, FOR ... NEXT); 5) avoid GOTO instructions; 6) use local variables as much as possible; 7) don't reinvent the wheel; 8) the more sub-programs or sub-functions used the better (Feldman, 1988:403).

Program Documentation

Program documentation included developing both the internal documentation (program code) and external documentation (user's manual). The internal documentation is meant to stand alone and allow a user or programmer to examine the QuickBASIC code. The development of the internal documentation consisted of adding comment statements throughout the QuickBASIC code telling future programmers what the different sub-programs accomplish. This allows future updates to be made without having to remember or relearn what that sub-program does.

The external documentation is formatted in the form of a user's manual which is intended to serve as the primary interface between the user and the application software. The user's manual for BasePlot was developed using the four major steps identified in the documentation shortfalls section of Chapter one. These steps are defining the goal, identifying the audience, establishing a time line, and testing and reviewing the manual (Kraft, 1988:62-64).

In identifying the user audience it was decided to make the user's manual task-oriented. This will allow the ABC analysts to accomplish the tasks they want by telling them how to perform these tasks. Aside from making the manual task-oriented, the user's manual was made user-friendly in the following ways: 1) easy to find information through a task-oriented index; and 2) talking to the user and not the system (Martin, 1986:20).

This internal and external documentation is further supplemented by a tutorial (demonstration) that will help the user get up and running on BasePlot in the least amount of time. The tutorial is included in the user's manual and it was tested to ensure the user is able to perform the required steps to get BasePlot running.

Field Testing

To evaluate the program for ease of use, the assistance of an uninvolved party was solicited. Five volunteers from AFIT were selected to test the programs useability.

The test consisted of using input and output from TSARINA stored on a 5.25" disk. The test evaluated whether the user could load the program, walk through the menus, and get the desired results. Any problems encountered by the user were corrected and retested by the programmer and the user.

Validation by an ABO Faculty Expert

A faculty member who is an expert in the field of ABO and in the use of TSARINA was given a demonstration of the program and asked to critique the application. Any recommendations for improvement of the program were implemented and retested by the programmer and the expert.

Program Implementation

Following expert validation, the implementation phase began. The program is targeted for analysts currently using TSARINA and it was distributed to them for use.

Summary

This chapter described the methodology process used to modify and enhance BasePlot. The next chapter discusses findings during program design and analyzes good programming characteristics with respect to the original goal of developing an efficient data analysis system.

III. Finding and Analysis

Introduction

This chapter describes the structured programming process and analyzes the success the programmer achieved in incorporating the characteristics of structured programming in developing BasePlot. There is a brief assessment of the software used in BasePlot's development. Analysis of the steps used in the documentation is covered next followed by a discussion of the procedures used to create a user-friendly manual. The chapter concludes with an analysis of BasePlot with respect to the original goal of solving the specific problem stated in Chapter I..

Incorporation of Characteristics of Structured Programming

As was mentioned in Chapter II BasePlot was assessed using eight structured programming rules. For clarification the rules are repeated here. The rules used in analyzing BasePlot are as follows: 1) perform detailed tasks in the sub-programs or sub-functions; 2) the sub-programs or sub-programs should be easy to understand and maintain; 3) use top-down flow control; 4) constrain the programming procedures to the three basic structured constructs (assignment and computation, IF ... THEN or SELECT CASE, FOR ... NEXT); 5) avoid GOTO instructions; 6) use local variables as much as possible; 7) don't reinvent the wheel; 8) the more sub-programs or sub-functions used the better

(Feldman, 1988:403). The rules mentioned above will be explained in greater detail in subsequent paragraphs which discuss BasePlot's analysis.

The first rule of structured programming keeps the detailed work in the sub-programs or sub-functions. The original BasePlot program followed this rule and any modifications made to BasePlot also followed this rule. For example, the calculation of the targets' locations are accomplished in separate sub-programs. This allows the programmer to enter a single X-Y coordinate and check the BasePlot calculation for that single target. If an error is detected the sub-program can be corrected and retried. All detailed tasks are accomplished in sub-programs or sub-functions, thus allowing them to be tested individually.

The second rule forces the programmer to keep the code simple. The detailed tasks do not have to be difficult to read and understand. The original BasePlot program was simple to understand but it lacked comment fields identifying what particular areas of code accomplished. The modified BasePlot code has incorporated the use of extensive comment fields explaining what the computer code does. This will make future modifications easier to accomplish.

Top-down flow of control, the third rule, makes programs flow smoothly form the beginning of the main program to the end of the program (Feldman, 1988:403). This was difficult to accomplish with BasePlot because so many of the sub-programs and sub-functions are called from many locations in the main program. While top-down flow was not strictly followed throughout, BasePlot flows generally from the top down.

The fourth rule, using three basic constructs, was maintained in the original BasePlot program as well as the modified portions. By constraining the program to three constructs the modifications were easy to make and future upgrades and modifications will be easy as well.

The GOTO instructions were used on occasion in the original BasePlot code but not extensively. The use of the GOTO statement can cause the program to be confusing and it breaks up the top-down flow. The occasional use was needed to direct error problems encountered while running BasePlot. QuickBASIC does not allow errors to be corrected within sub-programs or sub-functions and as a result any detected errors have to be corrected by the user in the main program.

Using local variables, the sixth rule, reduces the total number of variables needed and reduces inadvertent variable updating. A local variable is a variable that is used only when a sub-program or sub-function is used and therefore, variables with the same name can be used in different sub-programs or sub-functions. For example, the variable filename$ has a value of demohit.$$$ in one sub-program but filename$ has the value demoattack.$$$ in another sub-program. In this example the variable filename.$ is used twice in different sub-programs but the programmer only needs one variable name. Applying this rule also prevents variables from being accidentally modified in the main program.

The seventh rule pertains to using sub-programs or sub-functions which have already been shown to work in other programs in your program. This rule did not apply to BasePlot because there are no programs similar to BasePlot as was shown in Chapter I.

BasePlot uses 92 sub-programs or sub-functions and therefore rule eight has been used effectively. By applying this rule the programmer keeps the main program short and simple and easier to understand. The numerous sub-programs and sub-functions made the internal documentation process an easier task. This will also make future modifications simpler because the new modifications can be tested separately as sub-programs or sub-functions.

BasePlot definitely conforms to the structured programming philosophy. This makes modifications easier accomplish and enhances its use by ABO analysts because BasePlot can be changed to meet new user needs without a lot of difficulty.

Software Selection

QuickBASIC 4.5 was the software used in the original BasePlot program and to maintain consistency the modifications were programmed in QuickBASIC as well. QuickBASIC's earlier versions, BASIC and BASICA, did not allow sub-programs or sub-functions and could not follow the structured programming philosophies. However, QuickBASIC allows programmers to use structured programming. As was mentioned, earlier versions of QuickBASIC did not allow sub-programs or sub-functions and as a result everything had to be programmed in the main program. This makes updates cumbersome and difficult. This was the programmer's first QuickBASIC programming effort. But, QuickBASIC's ample menus and help facilities make QuickBASIC user-friendly and as previously mentioned it follows the structured programming rules. This made the task of modifying and enhancing BasePlot easier.

Incorporation of Documentation Steps

The external documentation (user's manual) was accomplished using the five major steps. The steps, as were addressed in Chapter I, are defining the goal, identifying the audience, establishing a time line, and testing and reviewing the manual (Kraft, 1988:62-64).

The goal of BasePlot's user's manual was to ensure the manual could be used by the inexperienced ABO analyst as well as the experienced ABO analyst. This was accomplished by adding additional information for the inexperienced ABO analyst. The additional information included a brief history of ABO and BasePlot, a tutorial (demonstration), and an ABO glossary of common ABO terms. The experienced ABO analyst can skip the extra information and read only the demonstration to understand BasePlot's capabilities. The experienced ABO analyst understands ABO planning so they need only the information required to improve their ABO analysis using BasePlot.

The intended audience of BasePlot is both the new ABO analyst as well as the experienced ABO analyst. BasePlot was originally developed for a new ABO analyst to help understand the inputs and results of TSARINA. This remains the major intended audience of BasePlot; however, BasePlot's features also allow the experienced ABO analyst to improve the ABO analysis they perform.

The time line for development of the user's manual was driven by the thesis requirements. The manual was completed three months prior to the thesis completion date to allow for testing and review by five volunteers.

The test and review revealed only minor flaws in the demonstration. The volunteers did not find any problems with the BasePlot program itself. The problems arose in the technical description of the directions in the demonstration. The corrections were made to the directions and retested by the volunteers. There were no other problems noted by the volunteers on the retest. The test also revealed some system limitations. BasePlot can not be used on a system that has less than 640K RAM memory. This is not a problem because the standard issue computer used in the Air Force has the required minimum.

Incorporation of User-Friendliness in the User's Manual

There are two areas of primary concern with making a user's manual user-friendly. The two areas are having a task-oriented index and ensuring the manual talks to the user. The index is comprehensive and geared toward tasks the user will perform when using BasePlot. This allows the user to look for help in the index if they are having trouble performing a menu option in BasePlot. The testing of the user's manual verified that this goal was met. The volunteers were asked questions about BasePlot and directed to find the information using the index. There were no problems encountered by the volunteers in answering the questions.

To test whether the manual talked to the user, two of the volunteers chosen had very little computer experience and they were asked to critique the ability of the manual to lead the user through the demonstration. Both of the volunteers said they felt as if the manual talked to them when they performed the demonstration. In fact one of the volunteers commented that the manual seemed to be written with the inexperienced computer user in mind. He continued by stating that there was no technical jargon confusing the demonstration.

Resolving the Specific Problem

Stated in Chapter I, the specific problem was to develop a software package (BasePlot) which could be distributed to Air Force ABO analysts. To obtain this goal the program needed to be modified, enhanced, documented, and assessed. The original BasePlot program needed only minor modification and enhancement. The majority of the work focused on complete documentation of BasePlot. The volunteers stated that BasePlot will improve an ABO analysts ability to perform simulated attack scenarios using BasePlot and TSARINA.

During the expert review and assessment by the ABO faculty expert, the original problem statement was explained and the instructor was asked if he believed application met the goal of preparing BasePlot for distribution. It was his opinion that the goal of solving the specific problem had been achieved (Halliday, 1990).

Summary

The completion of BasePlot for distribution has enabled TSARINA to be used in a more efficient and effective manner. Given the programmer's limited experience with structured programming and documentation development, BasePlot meets the ABO analysts need of having a visual representation of TSARINA's inputs and results. Chapter IV summarizes the results of the application of BasePlot, identifies BasePlot's strengths and weaknesses, and makes recommendations to improve BasePlot's usefulness.

IV. Conclusions and Recommendations

Introduction

This chapter summarizes the results of BasePlot's application, discusses the strengths and weaknesses of BasePlot, makes recommendations for further enhancements to be used for follow-on studies.

Results of BasePlot's Application

One of the biggest advantages BasePlot provides the analyst is a picture of the air base they are attacking. The pictorial representations can be used to brief the particular ABO situation at an air base. In the past the ABO analyst could only provide the statistical results of the attack provided by TSARINA; now they will be able to provide pictures showing the resultant attack. The pictures further emphasize the importance of a strong ABO program at our air bases by allowing ABO analysts the ability to show commanders on paper the air base's vulnerabilities.

The application of BasePlot also simplifies the ABO analyst's job. After developing the target and attack cards, the ABO analyst can use BasePlot prior to running the inputs through TSARINA. This way the analyst will catch obvious errors and prevent unnecessary repeated runs in TSARINA. For example, the ABO analyst may have intended to attack the runways, taxiways, and aircraft shelters and upon review in BasePlot the analyst finds the attack cards are off target. The ABO analyst can fix these errors as well prior to running TSARINA. The application of BasePlot will save the ABO analyst valuable time when performing ABO analysis.

The picture BasePlot provides will also help the ABO analyst look at different survival techniques (refer to Chapter I for a detailed list of survival techniques). For example, dispersing air base resources could save those resources dispersed from a particular attack. The ABO analyst can now run an attack using TSARINA and get a picture of the results using BasePlot. The ABO analyst can then disperse the resources by changing the target cards. Now, the ABO analyst can run the same attack as the first time and view the results on the air base's resources after dispersal. This same type of action can be applied to the other survival techniques.

BasePlot's Strengths and Weaknesses

One of the strengths of BasePlot is its ability to present TSARINA's inputs and results graphically. BasePlot provides a color representation of the air base on the computer screen. This overhead view of the air base can prove invaluable in addressing the problems facing an ABO program at a particular air base. For example, it is immediately evident if an air base has too many valuable resources grouped together simply by looking at the different colors.

Another strength of BasePlot is the ease with which it can be used. The enhancements made to BasePlot further ease the user's mind when they use BasePlot because they do not have to be concerned with making minor mistakes. The initial main menu of BasePlot allows the user to input an attack file, view the attack results, and zoom-in for a closer look. All of these steps can be accomplished with very few key strokes.

The user's manual is another strength of the software package. It makes using BasePlot even simpler. The user manual's demonstration decreases the amount of time it takes an inexperienced user to use BasePlot. The demonstration shows the user how to get from the DOS prompt and into BasePlot's main menu. The demonstration then runs through a quick session showing how to use BasePlot's features. Finally, the demonstration concludes by quitting BasePlot and returns the user to the DOS prompt.

The weaknesses of BasePlot are slow disk operations and the lack of specific target identification. The disk operations in BasePlot are intensive. These disk operations cause the screen updates to be slow. For example, every time you zoom in, BasePlot has to redraw the targets and attacks on the screen and it's a very slow process. The only way to overcome this is to use a RAM disk or a fast access hard disk. This weakness makes it impractical to use BasePlot on a floppy disk drive.

The second weakness of BasePlot is its inability to identify specific targets. It is easy to identify the runway, taxiway, air craft shelters, etc. because of the different colors. However, it is difficult to identify the difference between different maintenance facilities. This will be addressed further in the next section, future modifications and follow-on studies.

Recommendations for Modifications and Follow-on Studies

The following modifications are conceptual descriptors of the capabilities a graphics pre- and post-processor (BasePlot) to TSARINA should have to meet the needs of an ABO analyst. The first modification could be target database development using a digitizing tablet and QuickBASIC. The next modification could be attack database development on the screen using a mouse within BasePlot and attack database editing could be done on the screen as well. The final possible modification could be developing the ability for BasePlot to display specific target, attack, and impact point information on the screen.

Developing a target database using a digitizing tablet with a standard AF Civil Engineering base map would eliminate the need of writing the individual target cards currently required in TSARINA. This development would also include developing target continuation cards which specify how much of each resource (people, support equipment, vehicles, parts, etc.) is contained in each target.

Developing the attack database on the screen would eliminate the need of writing the individual attack cards currently required in TSARINA. After plotting a target database, the user would select an aim point using a mouse; a pop up window will then appear and allow the user to specify the weapon type and delivery parameters. Once everything is specified, the software will immediately plot the attack as a bomb stick (a line with circles) or a point impact (a circle). Color will be used to distinguish between weapon types. This option requires the development of weapon and delivery databases before implementation.

The editing of the attack database on the screen is similar to developing the attack database as described above. The user will select a plotted bomb stick or impact point using a mouse. Once selected, the user is then allowed to move it, change heading, change weapon type or delivery parameters, change stick length, etc. The software would immediately change the attack database and update the screen.

The final modification, target information, could be accomplished using the mouse. Using the mouse, the user will point to a target, bomb stick, or impact point. For targets, the program would display the facility/building number, target type, and any associated resources; if a weapon has impacted near the target, then the program would also display damage and casualty levels. For attacks, the program would display weapon type, delivery parameters, and so on. For impact points, the program would display weapon type, attack number and day.

These modifications will require addition sub-programs and sub-functions to be added to BasePlot's main menu. This is why structured programming is so important because these modifications can be added without drastically changing the main BasePlot program.

The author has heard student criticism of the thesis program not using "real-world" problems in the current curriculum. Any one of the modifications suggested in this thesis can be use as a project for a future thesis and the resulting thesis could be used by ABO analysts in the "real-world." By seeing immediate results of class projects or research activities, both students and DoD agencies would benefit.

Appendix

BasePlot User's Manual

by

Captain Richard M. Cockley

September 1990

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Background

BasePlot allows analysts experienced in the use of TSARINA (Theater Simulation of Air Base Resources INputs using AIDA) and a knowledge of Air Base Operability (ABO) planning to observe on screen the results of an attack scenario run in TSARINA. TSARINA is a Monte Carlo computer simulation model (Emerson, 1982:1) which assesses air base vulnerability to an enemy's conventional or chemical attack. TSARINA runs on a main-frame or micro-computer but it requires the user to have an extensive working knowledge of ABO. TSARINA gives analysts the opportunity to simulate attacks on various air bases but it does not provide any graphical representations of either its inputs or results. BasePlot provides these graphical representations. BasePlot was written by Captain Bob O'Neill, AFCSA/SAGO. It was edited and documented by Captain Richard Cockley, AFIT/LSG.

Overview

The purpose of the User's Manual is to introduce the user to BasePlot and its capabilities. Easeplot was designed using the QuickBasic 4.5 software program. However, the user does not need to have QuickBasic 4.5 loaded on the computer to install and run BasePlot.

BasePlot is an easy to use graphic pre- and post-processor. It includes nine different options for assessing simulated air base attack scenarios and thus determining the vulnerability of that air base. It also provides the ability to dump the screen contents to a DataProducts color printer in order to get a hard copy of the simulated attack.

BasePlot uses primarily three ASCII text files in the TSARINA data format: the target file, the attack file, and the hit file. These text files can be existing TSARINA files or built from scratch by an ABO analyst. The user has the capability of adjusting the data within these text files as long as they understand TSARINA's data requirements.

BasePlot allows the ABO analyst to assess air base vulnerability with these current text files. After reviewing the results of the simulated attack the ABO analyst can make changes to the TSARINA attack file, make another run using TSARINA, and then view the resulting hit file to get another assessment of the air base's vulnerability.

This User's Manual provides a brief history of ABO and BasePlot. The manual shows the user how to load BasePlot on their computer and then takes the user through a quick demonstration of BasePlot. Finally, the manual provides a detailed look at all the features of BasePlot.

ABO History

Since the beginning of aviation history, it has been the objective of all air forces to gain and maintain air superiority. The easiest way to achieve air superiority is to eliminate the enemies' ability to launch and recover aircraft. This strategy is addressed by Colonel Darrell G. Bittle, in "Air Base Survivability," which cites Italian Air Marshall Guilio Douhet as saying, in 1921, "it is more effective to destroy the enemy's aerial power by destroying his nest and eggs on the ground than to hunt his flying birds in the air" (Bittle, 1981:7). In order to combat the Soviet doctrine which emphasizes destroying the air fields of Western Europe a strong ABO program is required.

ABO was known as air base survivability (ABS) until 1983 when it was changed to include the ability to recover from attack as quickly as possible. Until this time the ABS program contained only those elements pertaining to surviving a hostile attack. This change increased the number of components in the definition of ABO. ABO is now defined as:

> an advocacy program aimed at improving our ability to survive and recover in a degraded environment and it includes those measures of active defense; base recovery; command, communications and control; and aircraft mods or enhancements necessary to defend against, mitigate the effects of, and recover from hostile enemy actions to achieve a sustained launch and recovery capability for effective combat operations. (Speigel, 1983:19)

The individual components (or pillars) of ABO include defense, survival, recovery, and generation (Figure 1 on the next page). The defense pillar is comprised of security radars, air interdiction aircraft, SAMs, point air defense (PAD), and air base ground defense (ABGD) (Ray, 1986:36).

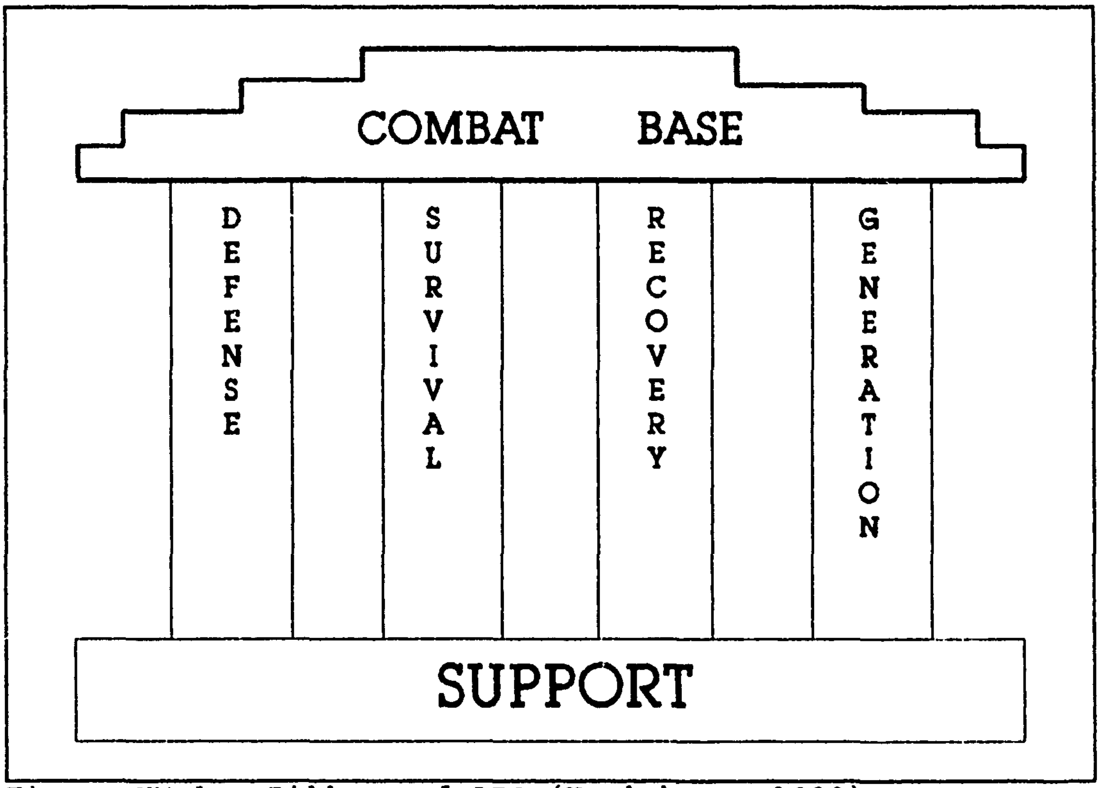

Figure UM-1. Pillars of ABO (Huelskamp, 1989)

Survival incorporates those actions taken by a base to deter the ability of the attacking force from destroying fixed facilities and resources to survive the attack. This survival pillar includes "hardening of shelters, protective construction, nuclear, biological, and chemical (NBC) defense capabilities, camouflage, concealment, and deception" (Nordeen, 1987:32-33). Recovery, the next pillar of ABO, includes assessment, rapid runway repair (RRR), disposal of unexploded ordinance (UXO), decontamination, and repair of critical facilities. The final pillar of ABO, generation, ensures the base is able to continue to generate sorties and includes Systems for Command, Control, Communications, and Computers (SC4) which ensures that command and control can be maintained during an attack and recovery from an attack (Ray, 1985:36).

The improvements to the conventional weapon systems of our adversaries highlight the importance of ABO and with upcoming budget reductions these issues will be brought to the forefront of the budget debates on Capitol Hill. The improved capabilities of conventional weapons forces the ABO analyst to simulate attacks on air bases that had been previously out of reach for our adversaries. The budget reductions ,on the other hand, will reduce the amount of funds allocated to ABO programs and force ABO analysts to look for ways to improve our ABO programs with less money. The reduced military budgets will also cause future exercises to be delayed or cancelled. These exercises provide valuable information about the ABO programs at our air bases and ABO analysts are going to have to rely more heavily on simulation programs to find out our air base vulnerabilities. Therefore, TSARINA in conjunction with BasePlot will help fill the void of the above mentioned problems.

BasePlot History

BasePlot was originally developed by Capt Bob O'Neill to help a new ABO analyst understand the complexities of the data entered into and received from TSARINA. TSARINA simulates planned attacks (attack cards) on air base shelters, facilities, surfaces, and resources (target cards) to determine actual weapon impact points (hit cards) and the associated damage caused by these weapons.

The target cards contain the data needed to place rectangular representations of shelters, surfaces, and other facilities (subsequently these will be referred to as just facilities) on the base in an X-Y coordinate system. Each individual target card has an X-Y coordinate representing the lower left hand corner of the facility. The target card also includes the length and width in feet. The final piece of data TSARINA needs in order to complete the facility is the orientation (heading in degrees). Using this information, TSARINA computes the remaining three X-Y coordinates to complete the rectangular representation of the facility. If for example the orientation is 0 degrees, TSARINA places the upper left corner directly north of the X-Y coordinate. The distance north, is equal to the width provided in the target card. The target card also specifies the target type (one of 30 different vulnerability categories), facility number, and other target descriptors.

The attack cards simulate an attack or series of attacks using up to ten different weapon types delivered by aircraft or surface-to-surface missiles. Each individual attack card contains the intended number of weapon-delivery passes (WDPs) an aircraft or missile would make on a base facility. This is identified by an aim point on the X-Y coordinate system. The next piece of information TSARINA needs is the heading, in degrees, of the WDP. The attack card continues with the number of bombs which will be dropped on that WDP and the length of the bomb stick associated with those bombs. The stick length varies depending on the aircraft delivery profile, the type of weapon, and the number of bombs in the WDP. For instance, a precision guided munition would have a stick length of zero while a drop of 10 general purpose bombs might have a stick length of 900 feet with 100 feet separation between each bomb. Finally, the attack card also specifies the type of weapon(s) in the WDP.

BasePlot transforms the numeric input data into graphical representations on a computer screen. The ABO analyst can then verify that the target cards were entered correctly by comparing the screen graphics to an air base map. For instance, if the base has a 12,000 feet long runway and, upon review of the targets on the computer screen it is only 1200 feet long, then the analyst can go back and change the appropriate target card to correct the problem. Similarly, the ABO analyst can also look at their attack data cards (BasePlot can display two attack files, each containing a maximum of 100 attacks) and visually check to see that the facilities they intended to attack were in fact attacked by the correct weapons in the correct sequence. Any discrepancies noticed can be fixed as well.

TSARINA normally produces statistical output showing the effect of the simulated attack on air base facilities. TSARINA can, in addition to statistical output, provide the ABO analyst with a hit file consisting of the X-Y coordinates of the individual bomb and unexploded ordinance (UXO) impact points for each TSARINA trial. Using BasePlot, the ABO analyst can then view the hits by attack, by trial, or by some combination of both to determine which assets on the air base are at risk from this type of attack or see if a minimum operating strip (MOS) exists. Furthermore, the ABO analyst can overlay the impact points on top of the aim points (attack data) to see how randomness and uncertainty affects the outcome of an attack. The ABO analyst can also assess if TSARINA adequately models the accuracy and reliability of the weapons.

System Requirements

The following is required in order to run BasePlot on your computer:

Hardware:	Zenith 248 or other IBM AT compatible computer, enhanced graphics adapter (EGA), and a minimum of 640K RAM available after loading DOS and other software.
Software:	MS-DOS or PC-DOS operating system, version 3.0 or later.
Recommended:	It is strongly recommended to increase speed and response to have an 80287 Math coprocessor, Virtual Disk or fast access Hard Disk.

BasePlot System Package

1 Diskette	Program Disk (1) Refer to Figure 2 on the next page to see the files included. The demo files on the disk are sample TSARINA files to demonstrate BasePlot.
Manual	This User's Guide

BP7_v2.Exe	- Executable Code (BasePlot Program)
DemoAtk.	- TSARINA ATT cards (attack aim points), text format
DemoAtk.$$$ & 1	- Binary attack files
DemoHit.	- TSARINA impact points (generated using) TSAR card option, text format
DemoHit.$$$ & 1	- Binary hit files
DemoTgt.	- TSARINA TGT cards (base facilities and structures), text format
DemoTgt.$$$ & 1	- Binary target files
DemoWpn.	- Weapon attributes (color and area or effectiveness), text format
DemoTA.	- Target attributes (color and fill option), text format

Figure UM-2. Program Disk Files

Installation Instructions

This section will tell you how to load BasePlot on your computer. Before you use the BasePlot program diskette, you need to make a "back-up" or "working" copy of your diskette. Follow the instructions in your DOS manual on how to do a disk copy. If you have a hard disk system, it is recommended that you install BasePlot on the hard disk because disk operations are intensive.

These installation instructions assume that the user is familiar with MS-DOS commands. To assist the user in the following guide, all key strokes required by the user are listed in **bold face type**. When typing in a response, the user may use either upper or lower case letters except where indicated to use CAPS only.

Dual Floppy Disk Instruction.

Step 1: Insure the MS-DOS diskette is in Drive A.

Step 2: Insert your "back-up" copy of the BasePlot program diskette into Drive B.

Step 3: Change to the B Drive by typing **B:**. At the "B" prompt type **bp7_v2** and hit **return**.

Step 4: You have reached the introduction screen of the BasePlot program and have completed the installation procedures. The screen should look like the one shown in Figure 3.

Hard Disk Instructions.

Step 1: Insure that MS-DOS has been installed on your computer.

Step 2: You need to create a directory for the BasePlot files. At the "C" prompt type **md baseplot** (or any other name you want to call the directory).

Step 3: Insert the BasePlot Diskette in Drive A. At the "C" prompt change the directory to "A" by typing **A:**.
At the A: prompt type **copy A:*.* C:\Baseplot** (if the directory that your BasePlot is in is different use that directory) After the files have all been copied onto the hard disk, change the "A" prompt back to the "C" by typing **C:** and then **return**. Remove the BasePlot Diskette from Drive A and store it in a safe place.

Step 4: Get into the BasePlot directory by typing **cd\baseplot** and wait for the directory to change. At this point type **bp7_v2** and hit **return** key to get to the introduction screen of BasePlot.

Step 5: You have reached the introduction screen of the BasePlot program and have completed the installation procedures. The screen should look like the one shown in Figure 3 on the next page.

BASEPLOT

Written by: Capt Bob O'Neill, Autovon 227-6520

Mobility and Operability Division
Directorate for Theater Force Analysis
Air Force Center for Studies & Analysis
(Edited & Documented by: Capt Cockley, AFIT/LSG)

This program is the property of AFSCA/SGAO; permission is granted to the user to make copies and distribute this program as long as this notice is included. While the author believes the program is accurate and reliable, the user assumes sole responsibility when using it.

PRESS ANY KEY TO CONTINUE ...

Figure UM-3. BasePlot's Introduction and Disclaimer

CHAPTER 2

BASEPLOT DEMONSTRATION

Getting Started

This chapter walks you through some of BasePlot's features. Please follow the instructions in this demonstration. If you make a mistake refer to Chapter 3 (BasePlot Advanced Features) for additional information. You should have completed the installation instructions in Chapter 1 before proceeding further. This manual will show you the actual screen displays and utilize examples to demonstrate how the program works.

The demonstration starts at the introduction screen and shows how to use BasePlot. Refer to Chapter 1, installation instructions, on how to get to the introduction screen. To continue, hit any key and the initial data screen will appear. It should look like Figure 4 on the next page.

At this point BasePlot is asking you for the name of the base the attack simulation will be performed on. You can enter any name, but for purposes of this demonstration, type **demo** and hit **return**.

BasePlot is now prompting you for the name of the target file. The target file contains the information BasePlot needs to plot the base facilities in an X-Y coordinate system. BasePlot will read either a text format or binary format (identified by a ".$" extension) of the target file. When reading a text file, BasePlot automatically creates the

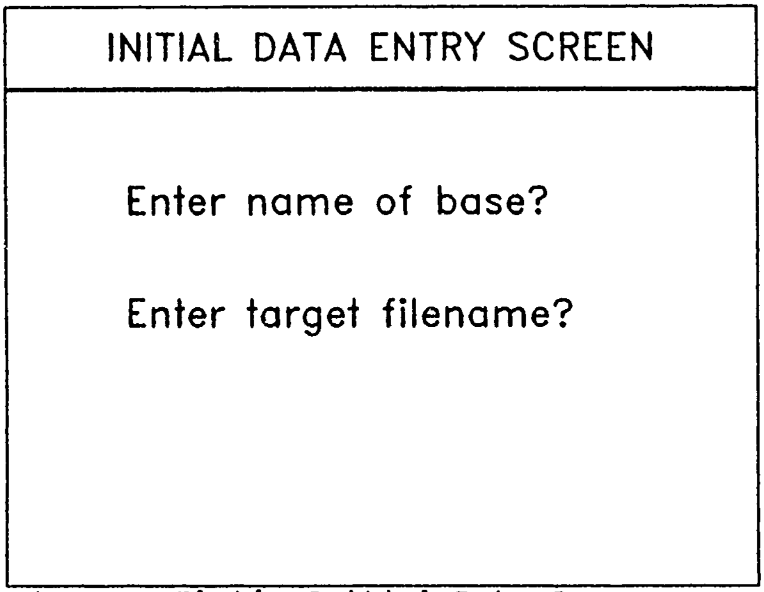

Figure UM-4. BasePlot's Initial Data Screen

binary file for subsequent use. Using the binary format significantly speeds up the reading process. Type **demotgt.$** and hit **return** (Note - if you enter the filename incorrectly be sure to include the .$ extension when you reenter the correct filename).

You have now reached BasePlot's main menu with the initial target screen. The screen should look like Figure 5 shown on the next page. The top line of the screen is the title line. This title line displays the name of the base followed by the files that are currently open. At this point the only file open is the demotgt file. The targets on the screen are color coded based on the target type (vulnerability category) specified in the target file. For example, runways, ramps, and taxiways are white. Refer to Figure 6 on the next page for a general description of the default target color settings.

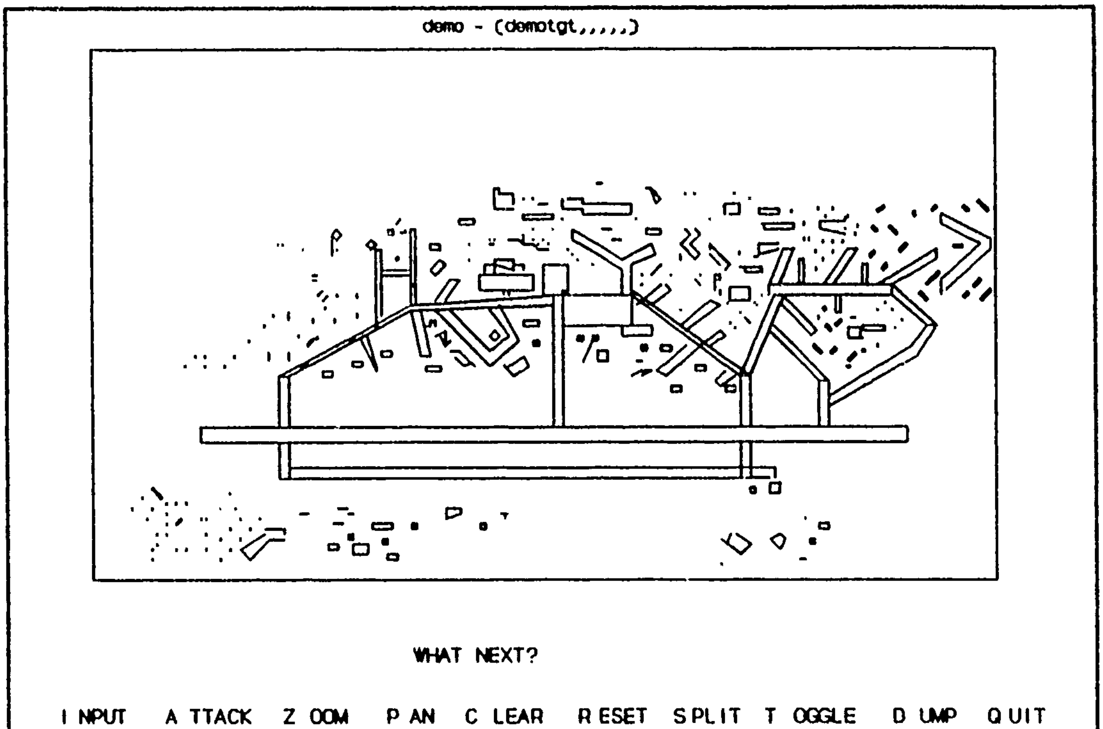

Figure UM-5. BasePlot's Main Menu

Target Colors

Type	Color	Resource
1	White	Runway
2	Green	Air Craft Shelters
4	White	Ramps, Taxiways, Electrical Power
5	Magenta	JP4 and Squadron Operatic
7	Brown	Shops and Maintenance Facilities
11	Blue	Munitions and Berms
25	Brown	Hangers and Dorms

Figure UM-6. BasePlot's Target Colors

The main menu, located on the bottom of the screen (refer to Figure 5), lists the following program options within BasePlot: I (Input), A (Attack), H (Hit), Z (Zoom), P (Pan), C (Clear), R (Reset), S (Split), T (Toggle), D (Dump), or Q (Quit) to return to the DOS operating system.

To choose an option, type the highlighted letter at the "What Next?" prompt.

At this point we will confirm that the runway is correctly entered. The runway should be 11,920 feet long and its lower left corner should be located at coordinates x=1,014 and y=1,940. The first thing you need to do is add a grid to the screen. This will assist in identifying coordinates.

To add the grid to the screen type t (Toggle) and then g (Grid). BasePlot has now added a grid starting at (0,0) and going to (14020, 8272), which is the maximum coordinate value for this target file, or the upper right corner. With the grid superimposed on the base we can now verify the runway coordinates. Looking at the screen we find the runway is correctly positioned.

If there had been an error in any of the target information you could make changes to the target file. To change the target text file you would have to go off-line and adjust the target card corresponding to that resource. Before we continue we need to remove the grid from the screen. To do this type t (Toggle) and then g (Grid). The grid will now wipe off the screen.

We will now take a look at an attack file. The attack file includes the information BasePlot needs to display the simulated attack on the screen. To input the attack file type i (Input). BasePlot will display a secondary menu of input options and change the "What Next?" prompt to an

"Input?" prompt. Type **a** (Attack) for an attack file. At the prompt "Enter ATTACK filename:" type **demoatk.$** and hit return. Similar to reading target files, BasePlot will read a text or binary format file and automatically create a binary file from the text data; reading the binary format significantly speeds up the process.

BasePlot opens the attack file and reads record pointer information into memory. This changes the title line to include the demoatk filename in bright white letters signifying it is the active attack file. BasePlot permits you to use two attack files at the same time but you can only display one at a time. Further information about toggling between active files is in Chapter 3.

We will now take a look at the attack that is currently active. To do this type a (Attack). BasePlot then prompts you for the attack number you want to view. BasePlot will load a maximum of 100 attacks per attack file; each attack consists of one or more WDPs. The demoatk file only has eight attacks in it. Type 1 and hit **return**. This causes BasePlot to draw attack #1 on the screen. The different colors represent different weapon types (refer to Figure 7 on the next page for the default settings). The weapons are represented by either a bomb stick (a line with circles) or a single aim point (a circle); both representations have a small T at one end to indicate heading of the aircraft on the WDP. BasePlot also adds a sub-title line giving you

Weapon Colors

Type	Weapon #	Color	Intended Resource
General Purpose Bomb	1	Light Green	Runway/Air Craft Shelters
Chemical Weapon	4	Light Red	Support Facilities
Runway Cratering Bomb	5	Yellow	Runway
Precision Guided Bomb	7	Bright White	Support Fac./Runway
Mines	8	Light Cyan	Support Fac./Runway

Figure UM-7. BasePlot's Weapon Colors

information about the displayed attack (F1 = File 1, A1 = Attack 1, D1 = Day 1, and 730 = Time of day).

Attack #1 was aimed at the runway, aircraft shelters, and maintenance facilities. We can see the runway cratering bombs (Yellow) and mines (Light Cyan) were aimed at the runway and were on target. The general purpose bombs (Light Green) were aimed at the aircraft shelters and they were also on target. Again, if any discrepancies existed you could make changes to the attack file by adjusting your aim points to ensure you attacked the facilities intended.

With attack #1 still drawn on the screen, we will now look at the hits associated with this attack. The hits are determined by the TSARINA Monte Carlo simulation model. The hit file is an output from TSARINA with the information BasePlot needs to plot the impact points on the screen. To view a hit file, type **i** (Input) and then **h** (Hit File). BasePlot then prompts you for the hit filename, type **demohit.$** and hit **return**. Similar to reading target files, BasePlot will read a text or binary format file and automatically create a binary file from the text data; reading the binary format significantly speeds up the process. Baseplot opens the hit file and reads record pointer information into memory. You will also notice BasePlot updates the title line to include demohit in bright white letters signifying it is the active hit file.

To view the results in the hit file type **h** (Hit). BasePlot prompts you for the attack number, type 1 and hit **return** to view the hits associated with attack #1. The hits are now plotted as solid circles on the screen and the subtitle line is updated with the hit file information (F1 = File 1, A1 = Attack 1, and T1 = Trial 1).

In order to get a closer look at the hits associated with the runway we need to pan down and zoom in. Before we pan down and zoom in we need to add the grid back on the screen, type **t** (Toggle) and then **g** (Grid). This will help understand what BasePlot does when it pans down and zooms in. First, we will pan down by typing **p** (Pan) and then **d** (Down). Baseplot will then pan the view down 1500 feet. After we pan down we can see the coordinate system has now changed. The new lower left corner is (0 , -1500) and the upper right corner is (14020 , 6772). Now, we need to zoom in, type **z** (Zoom). BasePlot prompts you for In, Out, Change ZF (Zoom Factor). Type **c** (Change ZF) and then 5000 followed by **return**. This changes the default value BasePlot uses when it zooms in and allows you to zoom to different parts of the base quicker. BasePlot has returned you to the zoom sub-menu. We will now zoom in with the new zoom factor by typing i (In). After updating the coordinate system BasePlot returns you to the main menu. The first thing you will notice is that there is no hit associated with the intended mines (Light Cyan) aimed at (2000 , 7000). This is a result of TSARINA deciding a malfunction occurred with that weapon. The malfunction could be an UXO, a dud, or a weapon that broke before reachng its destination. We can determine if it is a UXO by typing t (Toggle) and then u (UXO). This will turn on the UXOs and show them as empty circles. We can see that in fact this was an UXO. We need to get back to our original opening screen. To do this quickly you type r (Reset) and then s (Startup Coordinates). This will return the screen coordinates back to (0, 0) and (14020 , 8272).

If you wanted to look at some other attack/hits you can either leave the first attack on the screen or clear one or both attack and hits from the screen. We will clear both the attacks and hits from the screen. Type c (Clear) and then b (Both). This will erase the attacks and hits from the screen.

You have now completed the BasePlot demonstration chapter and you are ready to quit and return to the DOS operating system. Type q (Quit) to exit and you will be returned to the C:\baseplot\ or A:\ prompt. In Chapter 3 we will take a closer look at the individual features of BasePlot.

CHAPTER 3

BASEPLOT ADVANCED FEATURES

Introduction

This chapter is intended to provide you with a better understanding of BasePlot's features. Each section will cover the menu options available in BasePlot. The intention of the chapter is not to explain how BasePlot works but to describe what happens when you choose that option. If you want to change BasePlot's programed features you will need to understand how to program in QuickBasic 4.5. Documentation is available through the Defense Technical Information Center explaining the QuickBasic code.

Input

There are five input files used by BasePlot. They include: a target file, an attack file, a hit file, a target color file and a weapon color file. The target, attack, and hit files are either TSARINA-formatted ASCII text files or binary files created by BasePlot when reading a text file. The first two are TSARINA input files and the hit file is a TSARINA output file. The target and weapon color files are ASCII text files which change the default attributes BasePlot uses to display the targets and weapons. We will now take a close look at the TSARINA-formatted text files followed by a discussion of the color text files.

Target Files. The target file (demotgt) is the first file BasePlot uses to display TSARINA simulated attacks. This file contains TSARINA target cards which describe facilities on the base using X-Y coordinates. The target file is input by the user in the initial input screen prior to the main menu; it is not the target option on the input sub-menu. This option allows the user to input new target colors and is discussed later in this section.

The target cards contain the data needed to place rectangular representations of facilities on the base in an X-Y coordinate system. Each individual target card has an X-Y coordinate representing the lower left hand corner of the facility. The target card also includes the length and width in feet. The final piece of data TSARINA needs in order to complete the facility is the orientation (heading in degrees). Using this information, TSARINA computes the remaining three X-Y coordinates to complete the rectangular representation of the facility. The target card also specifies the target type (one of 30 different vulnerability categories), facility number, and other target descriptors.

The first time you read an ASCII text target file, BasePlot creates two additional files (filename.$$$ and filename.1) for subsequent use by typing filename.$ at the target filename prompt. Filename.$$$ is a random access/binary file of the original text data. Filename.1 is an ASCII text file storing the number of binary records and the startup X dimensions. Using the binary file option significantly speeds up reading the data.

If you enter an invalid filename, BasePlot will display an error message and ask for you to re-enter the information. If the file is not on the current drive, make sure you specify the entire path, filename, and extensions. If you make a mistake entering a filename without an extension, you must reenter the filename without an extension. If you make a mistake using a filename with the extension .$, you must reenter the filename with the .$ extension. For example, suppose you created a subdirectory on the BasePlot directory called demo and installed all your files there. Then you would type c:\baseplot\demo\demotgt.$ at the target file prompt.

To make changes to the ASCII text file you must use a file editor that allows you to import and save the file in ASCII text format. You also need to make sure you maintain the TSARINA column format. If you are not completely familiar with TSARINA's input format you should not try to modify the target file. When you make changes to the text file, you must re-read the text file in BasePlot to incorporate the changes in the binary file.

<u>Attack Files</u>. BasePlot uses the attack file to show the intended attack scenario. The ASCII text file contains TSARINA formatted attack cards to simulate an attack using up to ten different weapon types. The individual attack

card contains the intended number of weapon-delivery passes (WDPs) an aircraft or surface-to-surface missile would make on a base facility.

Each individual attack card contains the intended number of weapon-delivery passes (WDPs) an aircraft or missile would make on a base facility. This is identified by an aim point on the X-Y coordinate system. The next piece of information TSARINA needs is the heading, in degrees, of the WDP. The attack card continues with the number of bombs which will be dropped on that WDP and the length of the bomb stick associated with those bombs. The stick length varies depending on the aircraft delivery profile, the type of weapon, and the number of bombs in the WDP. For instance, a precision guided munition would have a stick length of zero while a drop of 10 general purpose bombs might have a stick length of 900 feet with 100 feet separation between each bomb. Finally, the attack card also specifies the type of weapon(s) in the WDP.

The first time you read an ASCII text attack file BasePlot creates two additional files similar to the ones created when reading a target text file (for more information refer to the target file section). The same guidelines for making target file changes also apply to making attack file changes.

Hit Files. BasePlot uses the hit file to show the results of the attack scenario input into TSARINA. A hit is the representation of the individual bomb and UXO impact point. This hit file gives the X-Y coordinates of the hits. Again, the same things that applied to the attack files also applies to the target files (for more information on the files refer to the target file section).

<u>Target Attribute File</u>. BasePlot can read an optional ASCII text file to change the default target color and fill attributes. When BasePlot draws a facility on the screen, it assigns a color and fill attribute based on its target type. BasePlot recognizes 30 target types or vulnerability categories and assigns each a default color and fill attribute.

DemoTA can by modified by any software package as long as it allows you to import and save the file in the ASCII text format. The first column contains the target type (see TSARINA manual). The second column contains the color attribute, an integer value between zero and seven (refer to Figure 8 on the next page). The third column contains the fill attribute: either 0 - not filled with color, or 1 - filled with color.

<u>Weapon Attribute File</u>. BasePlot reads an optional ASCII text file to change the default weapon colors and radii of effectiveness (the effect of the hit is represented by a circle within which damage would occur to facilities). When BasePlot draws attacks or hits on the screen, it assigns a color and radius of effectiveness based on the weapon type.

Target Color Attributes

0 – Black	3 – Cyan	6 – Brown
1 – Blue	4 – Red	7 – White
2 – Green	5 – Magenta	

[illegible] …or Attributes

BasePlot recognizes ten weapon types per attack/hit file.

DemoWpn can by modified by any software package as long as it allows you to import and save the file in ASCII text format. The first column contains the weapon type (see TSARINA manual). The second column contains the color attributes, an integer value between nine and fifteen (refer to Figure 9 on the next page). The third and forth columns contain the X and Y dimensions for weapons effects.

<u>Attack</u>

The attack option can be selected from the main menu. After making the selection, BasePlot asks the user which attack to display. BasePlot then draws the attack on the screen. You can enter the attack number you want to view or enter zero to see all the attacks. BasePlot draws the attacks using either bomb sticks (a line with circles) or

Weapon Color Attributes

9 – Blue	12 – Red	14 – Yellow
10 – Green	13 – Magenta	15 – White
11 – Cyan		

Figure UM-9. Weapon Color Attributes

single aim points (circles); both representations have a small T at one end to indicate heading. The different colors represent different weapon types.

<u>Hit</u>

The hit option can be selected from the main menu. After making the selection, BasePlot asks the user which attack and trial to display and then BasePlot draws the impact points on the screen. You can enter an attack number and trial to view the hits associated with a single attack; you can enter a zero for the attack number to view the hits associated with all attacks on a given trial; you can enter a zero for the trial number to view the hits associated with all trials of a given attack; or you can enter zero for both the attack and trial numbers to view all the hits associated with all attacks and all trials. BasePlot draws the hits using a solid circle. The hit (impact point) is determined by TSARINA and output to the ASCII text hit file.

Zoom

The zoom option allows you to zoom in or out by a uniform amount. The zoom factor is automatically adjusted to avoid errors. You can change the zoom factor (default = 2500 feet) by using the change zoom factor option on the zoom sub-menu. This increases the amount by which BasePlot changes the screen coordinates and allows the user to zoom to different parts of the base quicker.

Pan

The pan option allows you to pan up, down, left, or right by a uniform amount (pan factor). You can change the pan factor (default = 1500 feet) by using the change pan factor from the pan sub-menu. The pan factor controls the amount by which BasePlot changes the screen coordinates. For instance, if the pan factor is 2500 feet and the user pans down the coordinate system will change, the new lower left corner will be (0, -2500) and the upper right corner will be (14020, 4727).

Clear

The clear option clears the attack and hits from the screen. You can clear them individually or you can clea. both the attacks and hits at the same time.

Reset

The reset option allows the user to redraw the computer screen by providing a sub-menu of three options. The first option, view, redraws the screen without changing the coordinates; it corrects screen shifts caused by invalid keyboard entries. The match coordinates option resets the active view to match the inactive view coordinates; it is most useful when you have split the screen (refer to split) and you want the two screen coordinates to match. The final option available from the reset sub-menu is start-up coordinates; it resets the screen to the initial coordinates found when BasePlot draws the base on the screen for the first time.

Split

The split option splits the screen from 1 view to 2 views or vice versa. This allows you to look at two different attacks (or trials of the same attack) without overlapping them on the same screen. It also allows you to have a large scale and zoomed-in view at the same time.

Toggle

The toggle menu allows you to switch on or off many special effects.

Attack. This sub-option switches the active attack file. The active attack file is the one highlighted in white on the title line. You can only view on attack file at a time. Baseplot will allow you to load two attack files with a maximum of 100 attacks in each file. After you have input both attack files you can use this option to view the attacks in the different files.

Hit. This option works the same as the attack option.

Background. This option allows you to turn the background (target) colors on or off. Toggling (turning off/on) the background colors off allows you to see the attacks and hits more clearly.

Foreground. This option allows you to turn the foreground (attack and hit) colors on or off. Toggling foreground colors off and then using the function keys (refer to function key section) allows you to clearly see individual weapon types.

Grid. This option will toggle a grid onto the screen. The grid will assist in locating targets, hits, or attacks, and is automatically adjusted when zooming in or out.

Unexploded Ordinance (UXO). This option toggles the display of UXOs off. The UXOs will appear as empty circles on the screen. This will help you determine why some of the hits are not appearing on the screen.

Effects. This option will toggle the weapon effects on or off. Weapon effects appear as larger circles around aim points and impact points.

Screen. This option allows you to switch between active views. Although two views can be displayed on the screen simultaneously using the split screen option, only one view can be active at a time. You can also use this option to switch between active views when in the full screen mode.

Dump

This option dumps the screen to a DataProducts color printer. Since a variety of screen "grabber" software packages recognize many printers and plotters, BasePlot does not recognize any other device drivers. If you have a DataProducts printer you can print the contents in black and white or color. If you choose the color option, only attacks and hits are printed in color. .

Quit

This option returns you to the MS-DOS operating system.

Function Keys

The function keys toggle weapon colors on and off. For example, F1 toggles weapon type 1 on and off. However, since there are only seven color attributes for ten weapon types, some overlap is inevitable. As a result, when you press a function key to turn on or off a particular weapon type, you will actually turn on or off all the weapon types with that color attribute.

ERROR MESSAGES

Code	Error
5	An invalid CGA color
53	Bad filename of number. Allows the user to change the filename.
63	Bad record number
100	ReadNewHits: number of attacks > expected number of attacks. Stops execution.
101	RedNewHits: number of trials > expected number of trials. Stops execution.
102	ReadNewAttacks: number of attacks > max number of attacks. Stops execution.
103	ReadNewHits: Attacks are out of sequence. Stops execution.
104	ReadNewHits: Trials are out of sequence. Stops execution.

GLOSSARY

AGE	Aerospace Ground Equipment and other support equipment used for carrying out various tasks
AIDA	Air base Damage Assessment model; the forerunner of TSARINA
AIS	Avionics Intermediate Shops; special test equipment use for repairing avionic LRUs and SRUs
AMU	Aircraft Maintenance Unit; the organization providing maintenance for an aircraft
ATC	Air Traffic Control
BKEP	Ballistic Kinetic Energy Penetrator
BLSS	Base-Level Self Sufficiency stock of aircraft spare parts; composed of the stocks for peacetime, plus additional material to meet wartime demands
CAP	Combat Air Patrol
CAS	Close Air Support
CBU	Cluster Bomb Unit
CILC	Centralized Intermediate Logistics Concept
CIRF	Centralized Intermediate Repair Facility
COB	Collocated Operating Base
COMO	Combat Oriented Maintenance Organization
CONUS	Continental United States
CRS	Component Repair Squadron; q wing-level organization responsible for parts repair
CW	Chemical Warfare
EMS	Equipment Maintenance Squadron; a wing-level organization responsible for equipment maintenance and repair
FRAG	FRAGmentary order that specifies flight requirements
GP	General-Purpose bomb

ILM	Intermediate Logistics Maintenance; on-base parts repair supporting the AMU
IPE	Individual Protection Equipment for a chemical environment
LCOM	Logistics Composite Model
LRU	Line Replaceable Unit; an aircraft spare part with distinguishable subordinate components
MAE	Mean Area of Effectiveness
MMD	Mass Median Diameter of chemical droplets
MOB	Main Operating Base
MOPP	Mission Oriented Protective Posture (the CW ensemble)
NMCS	Not Mission Capable because of lack of Spare parts
NORS	Not Operability Ready because of lack of Spare parts; same as NMCS
NRTS	Not Repairable This Station
OST	Order and Ship Time in days; time for a NRTSed or condemned part to be replaced
PAA	Program Authorization, Aircraft
PGM	Precision Guided Munition
POL	Petroleum, Oils and Lubricants; often used as an abbreviation for aircraft fuel
POS	Peacetime Operating Stock; an organization's stock of aircraft spare parts for aircraft maintenance in peacetime
RAM	Rapid Area Maintenance; special mobile teams used for repairing aircraft battle damage
Resource class	All air base support resources are grouped into seven classes: personnel, equipment and AGE, aircraft spare parts, munitions, TRAP, building materials, and POL
Resource type	Difference types of resources may be distinguished with each resource class; e.g., different types of aircraft maintenance specialists

Resource packet	A user-specified percentage of a given resource type that is located at a specified target
RR	Flight line maintenance that removes and replaces malfunctioning aircraft parts with serviceable components; generally implies no local repair
RRR	Flight line maintenance that removes, repairs, and replaces aircraft spare parts (actually, usually removes and replaces with serviceable unit, and then repairs the malfunctions unit)
RRR	Rapid Runway Repair
SAMSOM	Support Availability Multi-System Operations Model
SCL	Standard Combat Load that designates the aircraft configuration and the mission dependent munitions to be loaded
SE	Support Equipment; usually referred to as AGE in TSAR
SRU	Shop Replaceable Unit; a component of an LRU
Target	A target is represented by a rectangle that is located in an X-Y coordinate system; individual buildings, runways, taxiways, parking areas, etc., can be designated as targets
Target complex	A target complex, such as an airfield or an industrial area, is a collection of rectangular targets
Target Type	A target type is specified for each target; all targets of the same type have the same vulnerability, and all resource types of the same resource class located at the same type of target have the same vulnerability
TBM	Tactical Ballistic Missile
TRAP	Tanks, Racks, Adaptors, and Pylons
TSAR	Theater Simulation of Air Base Resources
TSARINA	TSAR INputs using AIDA
UXO	Unexploded Ordnance
WRM	War Reserve Material
WRSK	Wartime Readiness Spares Kits

Index

<u>Bibliography</u>

Besch, Captain Edwin W., USMC-Ret. "Air Base Defense: New Solutions For the '80's." <u>Armed Forces Journal International</u>, <u>120</u>: 77-81 (May 1983).

Bittle, Colonel Darrell G., USAF. "Air Base Survivability." <u>Engineering & Services Quarterly</u>, <u>23</u>: 7-13 (Spring 1981).

Correll, John T., Senior Editor. "Air Defense From the Ground Up." <u>Air Force Magazine</u>, <u>66</u>: 37-43 (July 1983).

DiLullo, Joseph G. "Forward Air bases: Vulnerable but Survivable." <u>United States Naval Institute Proceedings</u>, <u>115</u>: 112-114 (January 1989).

Emerson, Donald E. and Louis H. Wegner. <u>TSAR User's Manual--A Program for Assessing the Effects of Conventional and Chemical Attacks on Sortie Generation: Vol. I, Program Features, Logic, and Interactions</u>. Contract F49620-82-C-0018. N-2241-AF. Rand Corporation, Santa Monica CA, August 1985a.

-----. <u>TSARINA--A Computer Model For Assessing Conventional and Chemical Attacks on Air Bases</u>. N-2244-AF. Rand Corporation, Santa Monica CA, August 1985b.

Feldman, Phil and Tom Rugg. <u>Using QuickBASIC 4</u>. Indiana: Que Corporation, 1988.

Fingerhut, Beverly. "Documentation Based on Faulty Assumptions. <u>ComputerData</u>, <u>11</u>:19-20 (October 1986).

Formwalt, Major Alex. "Air Base Design and Performance." <u>Instructional manual used in CIV ENGR 310, Computer Applications</u>. United States Air Force Academy, 1989.

Gornto, Cliff and Others. <u>ABO Model Comparisons CBAM/TSAR</u>. Contract F08635-89-C-0072. Orlando Technology, Inc., Shalimar FL, July 1989.

Guillemmette, Ronald A. "Application Software Documentation." <u>Journal of Systems Management</u>, <u>38</u>: 36-39 (May 1987).

-----. "What Readers Mean by Good Documentation." <u>Information and Management (Netherlands)</u>, <u>18</u>: 69-77 (February 1990).

Halliday, John M. Lt Col, USAF (Ret). Telephone interview. Rand Corporation, Santa Monica CA, 10 August 1990.

Huelskamp, Capt Robert M. and Others. Combat Base Assessment Model Version 1.0 Volume II, Analyst's Manual. Contract DNA001-85-C-0321. GRT-R-88-542. GARJAK Research, Inc., Solana Beach CA, July 1988.

Kraft, Melissa A. and Jean W. Pugh. "Designing a User Manual to Support an In-House Database." Database, 11: 62-64 (December 1988).

Law, Averill M. and W. David Kelton. Simulation Modeling and Analysis. New York: McGraw-Hill Book Company, 1982.

Martin, Peter. "Computer Documentation Problems." IEEE Transactions on Professional Communication, PC-29: 19-20 (December 1986).

Noble, Capt David R. Comparison of the TSAR Model to the LCOM Model. MS Thesis, AFIT/GLM/LSM/86S-54. School of Systems and Logistics, Air Force Institute of Technology (AU), Wright-Patterson Air Force Base OH, September 1986.

Nordeen, Lon. "Air Base Survivability--A Threat to NATO Air Power." National Defense, 21: 31-34 (September 1987).

Odescalchi, Esther Kando. "Documentation Is the Key To Success." IEEE Transactions on Professional Communication, PC-29: 16-18 (December 1986).

O'Neill, Capt. Telephone Interview. Air Force Studies and Analysis, HQ USAF, Washington DC, 5 December 1989.

Ray, Richard B., Rep. "Let's Protect the Platforms We've Paid For." Armed Forces Journal International, 123: 34-40 (October 1985).

Speigel, Wayne. "Air Base Survivability." Airlift, 5: 19-21 (Summer 1983).

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	September 1990	Master's Thesis

4. TITLE AND SUBTITLE
AN AIR BASE VULNERABILITY ASSESSMENT ANALYSIS TOOL FOR U.S. AIR FORCE WAR PLANNERS VOLUME I: DEVELOPMENT AND USER'S MANUAL

5. FUNDING NUMBERS

6. AUTHOR(S)

Richard M. Cockley, Captain, USAF

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology, WPAFB OH 45433-6583

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GLM/LSM/90S-12

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The purpose of this study was to improve the pre- and post-processor BasePlot before distributing it to experienced Air Base Operability (ABO) analysts and planners. Volume I contains four chapters and an appendix. Chapter I is the introduction containing the general issue, a statement of the specific problem, investigative questions, background of current issues on ABO, history and capability of TSAR and TSARINA, comparison to comparable software packages, BasePlot's capabilities, and a discussion of the current philosophies of documentation development. Chapter II details the thesis methodology used to solve the problem. This chapter describes software modification and testing, documentation, user testing, validation, and implementation. Chapter III describes the design, development, and outcome of the modified and enhanced BasePlot. Chapter III also provides an analysis of the pre- and post-processor BasePlot with respect to the goal of creating an efficient software package. Chapter IV describes the results of BasePlot's application and its strengths and weaknesses. This chapter also provides recommendations for follow-on research. The Appendix is BasePlot's User's Manual which documents BasePlot's operations and provides examples of how to use the software package.

14. SUBJECT TERMS Computer Programs,
Vulnerability, Air Base Operability, Graphics Pre-Processor, Graphics Post-Processor, Air Force Facilities, Bomb Damage, Conventional Warfare, Simulation Models, Theses. (RWJ)

15. NUMBER OF PAGES 97

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL